þÿ Amidst Uncertainty or Not? : Decision-Making in Software Startups

Kemell, Kai-Kristian

Springer Nature Switzerland
2019

þÿ Kemell , K-K , Ventilä , E , Kettunen , P & Mikkonen , T 2019 , Amidst Decision-Making in Early-Stage Software Startups . in S Hyrynsalmi , M Suonranta , A Nguyen Duc , P Tyrväinen & P Abrahamsson (eds) , Software Business : 10th International þÿ Conference , ICSOB 2019 , Jyväskylä , Finland , November 18 20 , 2019 , P Lecture Notes in Business Information Processing , vol. 370 , Springer Nature Switzerland , Cham , pp. 369-377 , International Conference on Software Business , Jyväskylä , Finland , 18/11/2019 . https://doi.org/10.1007/978-3-030-33742-1_29

http://hdl.handle.net/10138/310167
https://doi.org/10.1007/978-3-030-33742-1_29

unspecified
acceptedVersion

Amidst Uncertainty -- or Not? Decision-Making in Early-Stage Software Startups

Kai-Kristian Kemell[1][0000-0002-0225-4560], Eveliina Ventilä[1], Petri Kettunen[2][0000-0002-2928-5885], and Tommi Mikkonen[2][0000-0002-8540-9918]

[1] University of Jyväskylä, Finland
[2] University of Helsinki, Finland
kai-kristian.o.kemell@jyu.fi
{petri.kettunen|tommi.mikkonen}@helsinki.fi

Abstract. It is commonly claimed that the initial stages of any startup business are dominated by continuous, extended uncertainty, in an environment that has even been described as chaotic. Consequently, decisions are made in uncertain circumstances, so making the right decision is crucial to successful business. However, little currently exists in the way of empirical studies into this supposed uncertainty. In this paper, we study decision-making in early-stage software startups by means of a single, in-depth case study. Based on our data, we argue that software startups do not work in a chaotic environment, nor are they characterized by unique uncertainty unlike that experienced by other firms.

Keywords: Software Startups, Entrepreneurship, Decision-making, Cynefin Framework, In-Depth Case Study.

1 Introduction

Despite being extensively studied [10], startups still lack an accurate definition. Various characteristics have been attributed to startups to differentiate them from other firms. Characteristics typically associated with startups include (1) highly reactive, (2) innovation, (3) uncertainty, (4) rapidly evolving, (5) time-pressure, (6) third party dependency, (7) small team, (9) one product), (10) low-experienced team, (11) new company, (12) flat organization, (13) highly risky, (14) not self-sustained, (15) lack of resources, and (16) little working history [8]. In software startups, the role of software in the final offering may vary from being the core product to merely serving as an enabler or support of the main business idea (e.g. Uber) [11].

Klotins [3] highlighted the (lack of) empirical evidence behind many of these characteristics, questioning the uniqueness of software startups in relation to failure rates, lack of software engineering (SE) experience, innovativeness, market-related time pressure, and lack of resources. To this end, decision-making is another area where little is known empirically about software startups [10]. Software startups are considered to work amidst uncertainty that has even been described as a chaotic [2, 7, 8].

To provide empirical evidence into this on-going debate on software startup characteristics, we study software startup decision-making in relation to the uncertainty attributed to software startup in this paper. We do so by means of an in-depth case study

executed in an ethnography-inspired fashion, utilizing the Cynefin framework [6] for data analysis. Specifically, we tackle the following research question in this paper:

RQ: Using the Cynefin framework, how can we characterize the context software startups operate in in relation to decision-making?

2 Background

Few studies focusing on decision-making in the context of software startups currently exist [10]. On the other hand, organizational decision-making is a well-established area of research spanning various disciplines from economic ones to psychology and SE. Areas of research related to it are similarly diverse (e.g., business intelligence).

Some of the research on decision-making in the area of New Product Development (NPD) can be considered related to this context. Software startups search for new business models [10], whereas conventional business organizations *execute* business models. Software startups also make various decisions regarding SE [8].

In a literature review spanning multiple disciplines, Krisnan and Ulrich [5] presented a list of decisions related to setting up a product development project. They list various higher-level decisions related to product development that an organization needs to make when starting a product development project. Many of these decisions (e.g., "which technologies will be employed in the product(s)?") are relevant for software startups, while some are far more relevant to more established companies, (e.g., "will a functional, project, or development matrix organization be used?").

Extant literature has focused on conventional firms. The argument used to justify studies into software startups is that they differ from conventional firms, making the findings of such studies not (fully) applicable to them. Thus, studies seeking to understand how and whether startups differ from conventional firms are useful in this area.

3 Research Framework: Cynefin

To analyze our data, we utilize an existing theory: the Cynefin framework. Cynefin (Fig 1) is a decision-making tool from the field of knowledge management and complexity science [6]. It a sense-making tool intended to help its users understand the current context they are in. It presents a typology for decision-making situations.

The Cynefin framework splits decisions into five domains. The domains are based on the assumption of order, i.e., perceived causality of cause and effect. Each domain contains characteristics describing decisions in that domain, recommended actions for decision-making in that domain, and what type of practices should be used in it.

For example, the *chaotic* domain is characterized by a lack of perceivable cause and effect relations, as well as time pressure. The recommended actions are act, sense, and respond. I.e., one has to act quickly in order to establish some facts (sense) after which one has to respond to the situation again. This continues iteratively until it is possible to exit the chaotic domain. Crisis management situations are typical examples of chaos. In addition to the four main domains, *disorder* refers to a situation where the domain is unclear, necessitating further analysis of the situation.

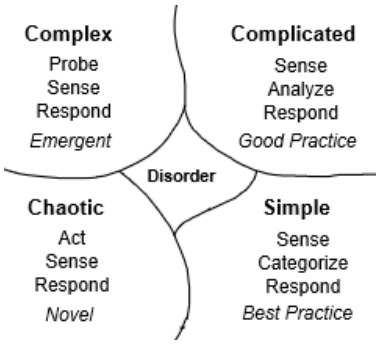

Fig. 1. The Cynefin Framework [6]

4 Research Methodology and Case Description

This study was carried out as a single in-depth case study, in an ethnography-inspired fashion. One of the authors worked as the founder of a software startup while collecting data. The case startup (from here on out Startup A) was founded in early 2018 by an inexperienced founder. Startup A produces a software service via a dedicated hardware solution: a wristband that would act as a replacement for business cards. Initially, it was unclear whether the company would develop only the software, or the hardware as well. The case startup is a real-life startup not founded to carry out this study.

The data collection started when the founder was the sole member of the team and the startup only had an idea to its name. Data from the case was collected from 30 April to 30 October 2018, in the form of video diary entries. Each evening, the founder produced a video recording detailing each decision they had made that day, or since the previous entry, along with detailing the current situation of Startup A. Occasionally, entries were produced less regularly, either because no decisions were made, or because the founder was not working on the startup e.g. during a weekend.

The recordings were later transcribed for analysis. These transcripts included a full transcript of each entry and a list of every decision discussed in each entry. From the transcripts, all decisions (136 total) were extracted into a list, which was then analyzed using the Cynefin framework (section 3). Each decision was evaluated and placed into the corresponding domain according to the criteria described in Table 1.

In order to increase the rigor of the analysis, we followed the protocol below:

1. Author A (founder) categorizes the data and provides reasoning for each choice.
2. Author B categorizes the data independently, without seeing A's analysis.
3. Author B compares the results of their analysis with those of Author A.
4. Decisions classified into the same category by Authors A and B are included for analysis (89 decisions, 65,4% of the total 136).
5. Author B studies the reasoning provided by Author A in the case of conflicting classifications (47 decisions, 34,6% of the total 136), and either changes their classifications by agreeing with Author A or continues to disagree.
6. Author C discusses the remaining conflicts with B (22. 16,2% of 136 total).

7. Remaining conflicts are classified based on the consensus of Authors B and C

Table 1. Criteria for Assigning Decisions into the Cynefin Main Domains.

	Decision speed	Effects observable	Potential decisions	Key action
Simple	Fast	Immediately or quickly	Usually one correct one	Categorize
Complicated	Slow	Quickly or slight delay	Multiple potentially correct ones	Analyze
Complex	(Very) Slow	Slowly	Numerous, difficult to choose good ones	Experiment
Chaotic	Fast	Immediately or quickly	No correct option, minimize risks	Act

5 Results

This section is split into five subsections according to the Cynefin domains. In our analysis, we highlight key observations in the form of Primary Empirical Conclusions (PEC), which we then discuss in the discussion section. Each subsection contains some examples of decisions and the full list of decisions is on FigShare[1].

5.1 Simple Domain

Out of 136 decisions, 36 were simple. Indeed, an early-stage startup faces many tasks that can be considered universal for small firms, such as setting up a company website and social media profiles, as well as creating a logo. A new firm, startup or not, has to carry out a large number of menial tasks.

Aside from such decisions relevant for virtually any startup, Startup A also made various context-specific simple decisions. These included setting up meetings with organizations belonging to the local startup ecosystem, dressing up for said meetings, and creating a calendar for the team in order to keep track of events.

However, many simple decisions led to further decisions that were complicated or even complex in nature. The decision to set up a website is simple because it is a best practice but deciding on what content to put on the website requires analysis.

5.2 Complicated Domain

Out of 136 decisions, 46 were complicated. These complicated decisions required capabilities from different areas of business and IT. For an early-stage startup with a small team, finding the required capabilities can be challenging, as was the case here. The inexperience of Startup A's team made many of the complicated decisions more

[1] https://doi.org/10.6084/m9.figshare.8298008.v1.

resource intensive. E.g., the team did not know how to find a limited company and thus had to devote resources towards studying how to.

PEC1: Inexperience increases the workload of a software startup team, contributing to the lack of resources typically associated with software startups

Indeed, many of the complicated decisions were related to resource allocation. The team constantly weighed between different funding options, trying to analyze which ones were the most likely to yield funding if applied for. As funding applications required time to prepare, this further aggravated the lack of resources.

PEC2: A lack of financial resources contributes towards a lack of resources in terms of person-hours.

Startup A was able to tackle some of the complicated issues with the capabilities they had. However, they occasionally had to learn new skills (designing), enlist outside help (video making), or hire new team members (programming). Issues related to team capabilities were prominent in Startup A in the complex and complicated domains.

5.3 Complex Domain

Out of 136 decisions, only 21 were complex (15,4%). Complex decisions require experimentation and cannot be consistently solved with existing good or best practices.

PEC3: Only a small portion of the decisions (very) early stage software startups make requires experimentation over analytical decision-making based on existing good or best practices

Many complex decisions made by Startup A were related to funding. Startup A was constantly balancing between different funding options with no clear way to determine the most likely successful one. The team was eventually forced to pursue funding options that would have yielded funding the fastest. Some of the sources of funding also had conflicting requirements (e.g. requiring a limited company (to not exist yet)).

PEC4: A lack of financial resources notably increases the level of uncertainty experienced by a software startup

Aside from funding, Startup A operated in a complex environment in relation to their service. Lacking hardware capabilities, they struggled to devise a technical MVP. With no technical MVP, they were forced to experiment with other ways of validation (e.g. surveys), and with no funding, they found it hard to experiment with existing hardware.

PEC5: A lack of technical know-how in the team is a critical issue for software startups and increases the uncertainty experienced by a software startup

5.4 Chaotic Domain

No decisions were categorized as chaotic. Startup A never experienced time pressure that necessitated acting without experimentation or analysis. Though they struggled with funding, they had never paid themselves salaries and thus it was simply their normal situation. It is, nonetheless, arguably possible for a software startup to find itself in a chaotic situation in relation to funding e.g. upon failing to pay salaries.

PEC6: Software startups do not operate in a predominantly chaotic environment

5.5 Disorder

Disorder is highly context-specific and difficult to identify retrospectively. Only some personal time management decisions of the founder were classified under disorder (e.g. whether to work weekends). Unforeseen events, fatigue, and one's own mood can affect such decisions, making it difficult to decide in advance what to do on such a high level.

6 Discussion

We have underlined our Primary Empirical Conclusions (PEC) in Table 5. In this section, we discuss each of them in relation to extant literature.

Table 5. Primary Empirical Conclusions Based on Analysis of the Data

#	PEC description (from analysis section)
1	Inexperience increases the workload of a software startup team, contributing to the lack of resources typically associated with software startups
2	A lack of financial resources contributes towards a lack of resources in terms of person-hours.
3	Only a small portion of the decisions (very) early stage software startups make requires experimentation over analytical decision-making based on existing good or best practices
4	A lack of financial resources notably increases the level of uncertainty experienced by a software startup
5	A lack of technical know-how in the team is a critical issue for software startups and increases the uncertainty experienced by a software startup
6	Software startups do not operate in a predominantly chaotic environment

We identified no chaos in the case startup (PEC6). This contradicts extant literature that has suggested that software startups operate in a chaotic environment in the context of Cynefin. One of the papers that originally suggested that startups operate in a chaotic environment dates back to 1998 [2]. While this may have been the case in 1998, we now understand startups better based on both practice and research. E.g., startups now have good practices to utilize and startup entrepreneurship is taught in universities.

Indeed, software startups also do not seem to operate under as much uncertainty as is typically attributed to them. Only 15,4% of the decisions of Startup A were considered complex, and to thus involve notable levels of uncertainty, while the rest could be solved with good or best practices (PEC3). However, not all decisions are equal. Some decisions may have much larger impacts on the future of the firm than others. Moreover, the case startup was a very early stage one still working on the first version of its service. This may not be the case in more mature startups that have progressed further. Finally, chaotic situations are arguably possible in software startups (e.g. if a startup that has already been paying salaries runs out of funding), even if none were present in the case startup. However, software startups hardly seem characterized by chaos.

In this study, most of the uncertainty experienced by Startup A stemmed from (1) lack capabilities in the team, and (2) lack of funding. Extant research has studied team

present from the start if they are required [9]. Our findings support this notion. However, problems related to securing technical know-how are not unique to startups[2].

The missing hardware capabilities quickly began to cause issues for Startup A as they struggled to validate their service idea, unable to develop an MVP. With no funding, they also found it difficult to experiment with hardware solutions. All in all, this resulted in an unreasonably long development cycle for an initial version of the product, which has been considered a key anti-pattern in software startups [4].

As for the lack of resources also experienced by Startup A, Klotins [3] recently argued that existing literature does not provide convincing arguments to support the uniqueness of startups in this regard. Our data does point towards software startups being unique in how they experience lack of resources (financial, person-hours). Lacking funding, startup A had to devote notable amounts of resources towards securing some (PEC4), which, due to their small team size, resulted in less resources being available for productive activities, e.g. programming (PEC2). A mature business would not task its developers with writing funding applications while lacking financial resources.

Finally, the inexperience of Startup A's team contributed to their lack of resources (PEC1). With no entrepreneurship experience, they were forced to devote resources towards e.g. studying different legal company forms. An experienced founder and team will arguably devote less time towards such menial activities, letting the team devote more resources towards productive activities. This supports extant literature which has linked business and technical experience with success in software startups [12]. Our findings help us better understand *why* this is the case.

6.1 Limitations of the Study

The single case approach is a limitation to the generalizability of the results. However, even a single case study can be enough to form a theory and is especially appropriate for new topic areas [1]. We do not consider our results conclusive and encourage further studies in the area. Moreover, data for this study were collected from the business-oriented founder, likely limiting the amount of SE-related decisions in the data.

We also underline three limitations in the Cynefin framework: (1) it is a decision-making framework for making sense of a situation rather than a categorizing tool for retrospective use as we have done here; (2) the subjective perception of an expert can make a complicated decision seem simple; and (3) the level of detail is important in categorization (e.g. deciding on applying funding vs. actually doing so). We have tackled the second limitation by conducting the analysis with three authors.

7 Conclusions

In this paper, we have conducted a single, in-depth case study of a software startup in an ethnography-inspired fashion. Based on our results, we argue that software startups

[2] E.g., https://yle.fi/uutiset/3-10669492 ("More than 10 000 open programmer positions, but no one to fill them").

are not characterized by a *unique* uncertainty, or chaos. The sources of uncertainty faced by the case startup (lack of financial resources and team capabilities, as well as idea validation) are issues any new or even mature business can face. However, a mature business might tackle these issues differently.

To summarize our findings into practical implications, we underline the importance of: (i) understanding what capabilities are needed in the team, and aiming to secure them early on; and (ii) inexperienced software startup founders understanding the need to study various practical entrepreneurship skills (e.g. how to find a limited company).

Finally, we encourage further research into what makes startups unique. Aside from uncertainty, various other characteristics have been attributed to software startups (see Introduction). Out of these characteristics, the uniqueness of startups in relation to failure rates, lack of (SE) experience, innovativeness, and (external/market) time pressure lack empirical support [3]. E.g. most startups fail, but so do most new firms [3].

References

1. Eisenhardt, K. M.: Building theories from case study research. Academy of management review, 14(4), pp.532-550 (1989).
2. Eisenhardt, K. M., Brown, S. L.: Time Pacing: Competing in Markets that Won't Stand Still. Harvard Business Review (1998).
3. Klotins, E.: Software start-ups through an empirical lens: are start-ups snowflakes? In Proceedings of the International Workshop on Software-intensive Business: Start-ups, Ecosystems and Platforms (SiBW) (2018).
4. Klotins, E., Unterkalmsteiner, M., and Gorschek, T.: Software Engineering Antipatterns in Start-Ups. IEEE Software (2019).
5. Krisnan, V., Ulrich, K.T.: Product Development Decisions: A Review of the Literature. Management Science 47(1), 1-21 (2001).
6. Kurtz, C. F., Snowden, D. J.: The new dynamics of strategy: Sensemaking in a complex and complicated world. IBM Systems Journal, 42(3), 462-483 (2003).
7. Nguyen-Duc, A., Seppänen, P., Abrahamsson, P.: Hunter-Gatherer Cycle: A Conceptual Model of the Evolution of Software Startups. Proceedings of the 2015 International Conference on Software and System Process, pp. 199-203 (2015).
8. Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., Abrahamsson, P.: Software development in startup companies: A systematic mapping study. Information and Software Technology, 56, pp. 1200-1218 (2014).
9. Seppänen, P., Liukkunen, K., Oivo, M.: Little Big Team: Acquiring Human Capital in Software Startups. In Proceedings of the 2017 International Conference on Product-Focused Software Process Improvement, pp. 280-296 (2017).
10. Unterkalmsteiner, M. at al.: Software Startups – A Research Agenda. e-Informatica Software Engineering Journal, 10(1), pp. 89-123 (2016).
11. Wang, X., Edison, H., Bajwa, S.S., Giardino, C., Abrahamsson, P.: May. Key challenges in software startups across life cycle stages. In Proceedings of the 2016 International Conference on Agile Software Development, pp. 169-182 (2016).
12. Zhang, J.: The advantage of experienced start-up founders in venture capital acquisition: evidence from serial entrepreneurs. Small Business Economics 36(2), 187–208 (2011).